Research in Engineering Design (1994) 6: 1–13


Research in
Engineering
Design

# A Model-Based Method for Organizing Tasks in Product Development

Steven D. Eppinger, Daniel E. Whitney, Robert P. Smith and David A. Gebala
Massachusetts Institute of Technology, Cambridge, Massachusetts, USA

**Abstract.** *This research is aimed at structuring complex design projects in order to develop better products more quickly. We use a matrix representation to capture both the sequence of and the technical relationships among the many design tasks to be performed. These relationships define the "technical structure" of a project, which is then analyzed in order to find alternative sequences and/or definitions of the tasks. Such improved design procedures offer opportunities to speed development progress by streamlining the inter-task coordination. After using this technique to model design processes in several organizations, we have developed a design management strategy which focuses attention on the essential information transfer requirements of a technical project.*

**Keywords.** Concurrent engineering; Design management; Design process improvement

## Introduction

Intense competition forces firms to develop new products at an increasingly rapid pace. This mandate places substantial pressure on engineering teams to *develop better products* and at the same time to *develop products faster.* Engineering organizations have responded to these two challenges by popularizing the concept of concurrent or simultaneous engineering.

Most of the literature on concurrent engineering describes a successful multi-functional team approach to product development. It is recommended that the multiple issues be integrated by allowing design engineers to work closely with manufacturing engineers, field service engineers, and representatives of others interested in the manufacture and use of the product [6, 15, 37]. There are numerous anecdotes describing small projects (five to ten people) successfully using this "team integration" approach. We call this *concurrent engineering in the small.* It works because small teams can work closely together and the challenging technical issues are exposed and resolved by mutual understanding.

In contrast, *concurrent engineering in the large* describes the challenges we see facing larger projects (hundreds of people) where the development effort is decomposed into many smaller projects, each possibly performed by a small team. The design of an automobile, aircraft or computer can involve thousands of engineers making millions of design decisions over several years. None of these many tasks is performed in isolation [5]. Each design choice may be a tradeoff affecting many other design parameters. Facilitating the transfer of information among design groups is an essential organizational task of product design managers [2, 7, 19, 35]. Their primary development challenge is to integrate the many sub-problem solutions into a well-designed system. Some firms address this by assigning system engineers or conflict resolution engineers to handle the interactions between sub-systems and to arbitrate the disputes between teams. The trouble is that such interactions are often poorly understood and are rarely known in advance.

Most product development entails the redesign of existing items, rather than design of entirely new items. We have observed that in most large firms there is a huge investment in existing design procedures, often heavily bureaucratized. While these procedures seem to work well, they may have grown up organically and historically. Without having been subjected to careful analysis, the internal inefficiencies or irrationalities remain largely undetected.

The goal of this research is to assist large concurrent engineering projects to achieve the benefits experienced by small teams which can discover technical interactions easily. We recognize that every issue in a large project cannot be considered simultaneously by a focused team. We have found that design managers need some tools to help structure projects effectively. To start, we explicitly model such inter-task dependence and use this knowledge to redefine and reorganize the engineering tasks.

*Correspondence and offprint requests to:* Steven D. Eppinger, Sloan School, E53-347, Massachusetts Institute of Technology, Cambridge, MA 02139, USA.

This work departs from traditional project management and engineering design research in two important ways. First, we consider relatively detailed models of the development procedures, which allow us to understand the complex interactions among activities. These models permit us to capture the basic iterative nature of design (whereas traditional project management models [39] utilize simple precedence network models that cannot depict iteration). Second, by explicitly modeling such coupling, we can attempt to reduce the complexity involved in large projects by restructuring entire development procedures (whereas most engineering design research is aimed at improving the isolated effectiveness of the more focused engineering tasks).

We aim to develop a design management strategy which can improve the product development process by:

1. Documenting existing procedures for scrutiny of the team.
2. Resequencing and regrouping design tasks to reduce complexity.
3. Sharing engineering data earlier and/or with known confidence.
4. Redefining critical tasks to facilitate overall project flow.
5. Exposing constraints and conflicts due to task interactions.
6. Helping design managers to place emphasis on task coordination.
7. Allowing design iteration to be strategically planned.

Our approach involves mapping an existing or proposed design procedure into a simple array representing the complex inter-relationships among the many design tasks which must be accomplished. The argument for a design methodology which corresponds to the underlying structure of the design problem has been articulated by Steward [32], Simon [25], and other authors, most notably Alexander in the 1960s [1]. The analysis we will perform considers the relative importance of each input to the design decisions, allowing the information requirements to determine the appropriate scheduling of the activities. The result of this analysis is an array of options for a manager or engineer to rearrange or reprioritize the tasks. Strategies include decoupling and resequencing tasks, insertion of new decisions points, splitting or condensing tasks, and other schemes to improve the flow of information and decisions.

## Sequencing Development Tasks

Creating a detailed information-flow description of a project involves explicitly mapping out the technical aspects of the design procedure. We contend that to be most useful, the design representation must include not only the sequence of the tasks but also the many technical and informational relationships among the tasks. The description we use is based on Steward's design structure matrix. However, before presenting this method, we will illustrate some general issues in sequencing design tasks and transferring engineering information with a simple example.

Consider two development tasks, labeled A and B. Figure 1 shows directed graphs (digraphs) [36] of three possible ways in which the two can be related. If task B simply requires the output of task A (or vice versa), then the two tasks are *dependent* and would typically be done in series. On the other hand, the two would be entirely *independent* if tasks A and B could be performed simultaneously with no interaction between the designers. Finally, if task A needs information from task B, and also task B requires knowledge of task A's results, then the two tasks are *interdependent*, or *coupled.*

Coordinating either the dependent (series) tasks or the independent (parallel) tasks is quite straightforward. Certainly with no limitation on resources, the parallel tasks can be completed more quickly. The interdependent (coupled) tasks are much more challenging to organize, often requiring much more design time and many iterations of information transfer [33].

To illustrate using a familiar theme, we can envision task A to represent a product design function, and task B to represent the associated manufacturing engineering function. Then our series model depicts the outdated "throw the design over the wall" methodology. The parallel tasks model might then represent an idyllic view of simultaneous engineering, where both design and manufacturing functions are given the same challenge, and they magically develop product and process concurrently (without complex interactions). The coupled tasks model is a more

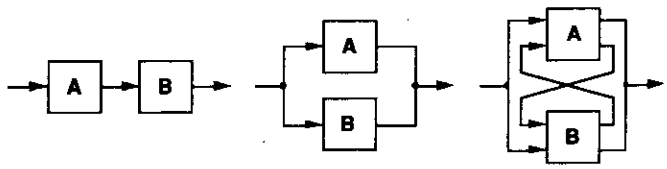



**Fig. 1.** Three possible sequences for two design tasks.

realistic diagram of simultaneous engineering, where the information transfer is essential and iteration is typical.

## The Design Structure System

Steward's design structure system [31, 32] uses a representation which allows the direct coupling of any task to another. Figure 2 shows the design structure matrix in its original binary form as described by Steward, where the design tasks to be performed are each represented by an identically labeled row and column of the matrix. The marked elements within each row identify which other tasks must contribute information for proper completion of the design. For example, the marks in row D are in columns E, F and L, indicating that execution of task D requires information to be transferred from tasks E, F and L. We would then desire these three tasks to be performed *before* task D. (The diagonal elements in the matrix are essentially meaningless at this point but are included to distinguish the diagonal and the upper and lower triangles of the matrix.)

The first step of design structure analysis is to find a sequence of these design tasks which allows this matrix to become lower triangular. If the tasks can be sequenced so that each one can be executed only after it receives all the information it requires from its predecessors, then no coupling remains in the design problem. However, this rarely happens. Instead, analysis usually yields a matrix in a block–lower-triangular form. The blocks on the diagonal depict coupling due to feedback, and the remaining below-diagonal marks represent the feedforward of information to later tasks. Figure 3 shows the matrix from Fig. 2 after the twelve tasks have been rearranged (partitioned) by interchanging rows and also swapping the corresponding columns to achieve a more organized design sequence.

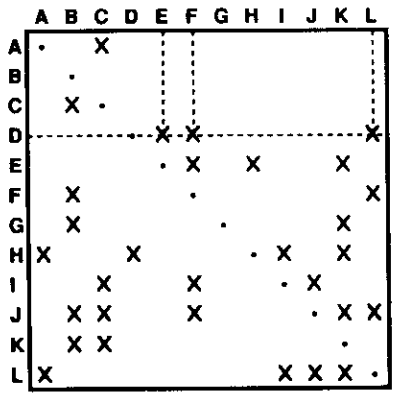



Fig. 2. A binary design structure matrix, unpartitioned.

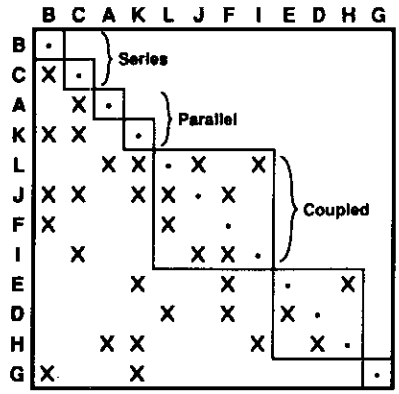



Fig. 3. The binary design structure matrix, partitioned to represent a sequence.

The partitioning process has sequenced the tasks to be performed in the order: B-C-A-K-L-J-F-I-E-D-H-G. The matrix shows that task C is dependent upon task B, so they are performed in the sequence B-C. Tasks A and K are both dependent upon task C but they can then be completed in parallel (since task K does not depend upon task A, and vice-versa). The two "blocks" encompassing the task sequences L-J-F-I and E-D-H identify two sets of coupled tasks, the most challenging aspects of this design problem. Each of these two sets of tasks must be performed simultaneously, and the information transfer required may take the form of iteration and/or negotiation.

The matrix partitioning in Fig. 3 is unique (in terms of grouping coupled tasks into blocks); only the ordering of parallel activities and sequencing within the blocks depend on the algorithm used to reorder the tasks. Several schemes for identifying the blocks are available, including techniques based upon binary matrix algebra [12], a rule-based (expert system) analysis [22, 23, 28], and Steward's loop tracing procedure [30]. We have also developed improved partitioning algorithms which are discussed in another paper [8].

When the design structure matrix cannot be manipulated into lower triangular form, we then seek a form that minimizes the size and number of the remaining blocks on the diagonal. Collapsing these blocks into single tasks (as would be required for PERT analysis) would certainly make the project *appear* to be simpler. In our example, we would combine tasks L, J, F and I into one task and then collapse tasks E, D and H into another. We would be left with seven tasks in lower-triangular form instead of the twelve tasks as shown. However, this approach hides the real design problems and precludes any opportunity to further improve the design procedure by applying other techniques.

Since the coupled blocks in the design structure matrix represent design iteration, choosing the proper sequence to work through even these tasks is quite important. We believe that there is significant advantage in performing the initial "guesswork" required to start the design iteration at a specific task which may allow the design to converge quickly. This can reduce the time required by the iterative process by isolating uncertainty and increasing the confidence associated with the design decisions. Several algorithms also exist for sequencing within these blocks. This problem is analogous to the analysis of chemical flow sheets [9, 10, 12]. Steward terms this procedure *tearing*, since the unknown information corresponds to elements being torn from the matrix to get the iteration started. Effective tearing requires detailed knowledge of the problem domain so that the less important elements are torn to leave the essential ones below the diagonal. (Note that tearing does not actually alter the matrix by removing any of the marks, rather these procedures simply find a suitable ordering within a block.)

An encouraging demonstration of Steward's matrix representation is found in recent work at NASA [22, 23, 28]. As an example problem to test their rule-based partitioning algorithm, they modeled the process of designing a complex spacecraft antenna system with over 50 interrelated tasks [17]. The design structure analysis showed that in this design problem there is a small number of large subsystems containing from 5 to 20 tasks each. These coupled groups of tasks are then performed in the sequence: actuators, sensors, structures, dynamics, controls, etc.

## Extensions to the Design Structure Matrix Representation

As presented by Steward, the binary design structure matrix represents only strict precedence relations. (A task either does or does not depend upon another task.) In complex design (sub-)problems, we find that the binary matrix is often crowded with weak dependences, and this leads to an extremely coupled design matrix. Furthermore, there is no representation in these models of task completion time, which could be used to understand overall project timing.

We therefore extend the basic representation by explicitly including measures of the degree of dependence and of the task durations, so that we can use more sophisticated analytical procedures to further improve the design process. Figure 4 shows a numerical design structure matrix which uses values

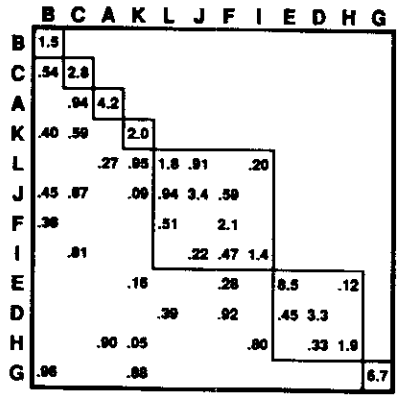



Fig. 4. A numerical design structure matrix.

in the off-diagonal positions to represent the relative importance of each task dependence. (Blanks are zeros, depicting no task dependence.) The diagonal values represent task completion time.

The rules for partitioning and tearing this matrix can now consider rearranging tasks to (for example) minimize the iterative backtracking required within the coupled sub-systems by arranging the more important feedback marks closer to the diagonal. For example in Fig. 4, the tasks are sequenced such that the lesser dependences lie further above the diagonal.

The numerical coupling values need not necessarily depict the strength of the task input dependence. Other metrics to consider include task communication time, functional coupling [21], physical adjacency, electrical or vibrational characteristics, parameter sensitivity, historical variance of task results, certainty of planning estimates, or volume of information transfer. Furthermore, each matrix element could instead be a vector of multiple measures, such as certainty and strength of dependence. Note that each of these metrics results in a different representation of the process, and appropriate analytical models would be required for each.

For example, if a task vitally depends on information from another task but that information is known to lie within predictable limits, then the dependent task might be able to start based on a good guess of the anticipated information. Thus the dependence would be represented as weak. Similarly, if the task depends only slightly on information that is largely unpredictable, the dependence might again be judged as weak. Contrarily, needed information with large impact and large variability implies a strong dependence. (We cannot start without it, nor can we predict it well enough.) An "importance ratio" can be calculated as the basis for determining the strength of the dependence. This ratio would be similar in definition to Taguchi's "signal-to-noise" ratio used to

compare the relative effects of parameters [4]. To sequence a group of coupled tasks more smoothly, we would begin with the one which is missing only information that is relatively certain. Such a strategy would reduce the number of design iterations necessary.

Developing a numerical design structure model can be quite difficult; however, we can suggest several schemes for finding the numerical values representing task dependence. The most straightforward method is to combine this step with the initial data-gathering phase where the dependences are identified in the first place. Alternatively, engineers can be interviewed specifically to find the coupling strength for the dependences just in the blocks. We have had success using both of these schemes. Other methods include analytical approaches where the values could be extracted from an engineering task–parameter sensitivity analysis, or experimental approaches where the values are found by testing. It is also likely that methods of constraint propagation [20, 29, 34] can be used to help write the matrix representation automatically. Note that in problems that are completely described by equations, the equations can be resequenced at will if numerical solution methods are acceptable [38]. That is, the equivalent of a lower triangular representation can always be found or simulated in such cases. In general, problems are only partially described by equations, and a mixed approach is required.

To extend this concept, we have developed two design iteration models, each based upon a different metric for task coupling and an associated set of assumptions about the underlying engineering information transfer. Both of these models are intended to predict iteration time and use task duration estimates in the diagonal positions. In the sequential iteration model, the off-diagonal numerical values indicate the probability that one additional iteration will be necessary if the interdependent tasks are performed in the specified order. The analysis supposes that within the coupled/iterative portion of a design procedure, tasks are performed one at a time. After each is executed, a probabilistic choice determines whether an earlier task is repeated or whether the process progresses on toward completion. In a paper describing this work [26], we explain the model and the Markov chain analysis used to determine iteration time.

The work transfer (parallel iteration) [27] model uses off-diagonal values which measure what portion of information produced during the first iteration would need to be changed during the second iteration. In this way the design process can be seen as a series of iterations of decreasing duration. We assume that work progresses simultaneously on all of the tasks and that they create rework for one another at various rates. We developed an analytical procedure for computing the rate of convergence of the subproblems comprising this coupled system of tasks. This approach involves analyzing the eigenstructure of the matrix and interpreting the mode shapes contained in the eigenvectors. Using the eigenstructure analysis, we are able to identify the most iterative subproblems (which we call design modes) within the matrix.

## Exploring Design Structure Data

We have used the design structure system to represent product development procedures at several different firms. In representing design process data, we have created two types of models: high-level task-based descriptions, where the inter-task information transfer relationships can be studied; and low-level parameter-based descriptions which document the technical interactions among the engineering parameters. The two examples given here are from the automotive industry; the first shows a task-level description, and the second example demonstrates a parameter-level description of a design problem. We have found both types of descriptions to be very useful and insightful in different ways; however, they are complementary to one another, and we will in the future attempt to collect enough detailed data to create hybrid models.

### Task-Level Design Description

Figure 5 shows one portion of the matrix representation for the design of a single powertrain component at General Motors. To develop this matrix, we began with the company's existing design process documentation, a set of existing IDEF diagrams which describe the "as-is" component design procedure. (IDEF is a standardized process-diagramming technique used extensively by US military contractors.) This modeling technique [13, 24] requires the model authors to extensively interview members of the design organization (at many levels) to characterize the relationships among the tasks. The legend in the figure identifies various task coupling labels. The marks I and C represent two different types of task dependence: input and control, which are defined by the IDEF methodology. Our interpretation is that the C marks depict smaller dependence than do the I marks. We have labeled some of the above-diagonal marks F to depict the feedbacks in the design procedure which drive iteration. (Most of the F marks were originally

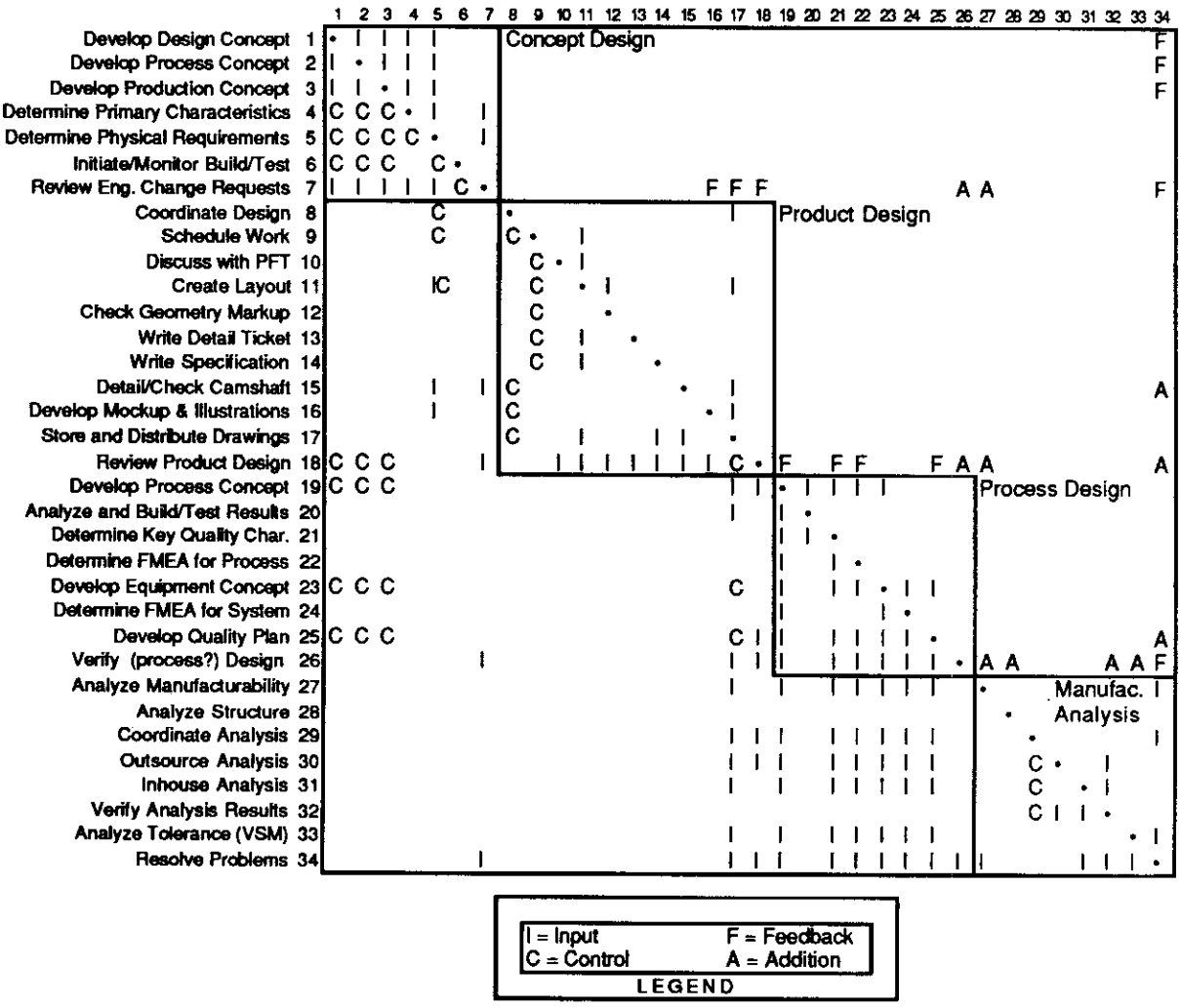



**Fig. 5.** Task-based design structure matrix for component design.

I marks in the original IDEF model, but are explicit feedback in this particular task sequence.) Finally, we have added the marks labeled A to improve the design process by providing additional paths for information flow which were not present in the IDEF model.

The design matrix in Fig. 5 appears in the exact sequence documented by the IDEF data. That is, we found the design tasks to be naturally partitioned into the block form shown, with four tightly linked blocks representing the major (iterative) design activities, coupled through only a few tasks. Application of the partitioning algorithms discussed above would offer minor changes in the task sequence. In this case we choose to study the existing process ordering to learn about its underlying structure. The organizational structure suggested by this matrix would be four design teams, each performing one of the major activities. In fact, that is the structure which is in use at the company, perhaps having evolved to minimize the number of external feedback loops.

Within each of these four blocks, the tasks are performed somewhat concurrently, as suggested by the strong coupling. However, the overall project flow is largely sequential. For example, the process design activity requires several inputs from the final stages of product design (store and distribute drawings). One insight drawn from this model is that to begin process design earlier, this dimensional information must be transferred sooner. The above-diagonal marks labeled as feedback represent paths of information transfer requiring potentially long-lead-time iterations, or "design rework" that is to be avoided if possible. We have found this sort of task-level description to be useful in exposing the many forms of internal coupling in complex development procedures.

To create a numerical task-based DSM, we have

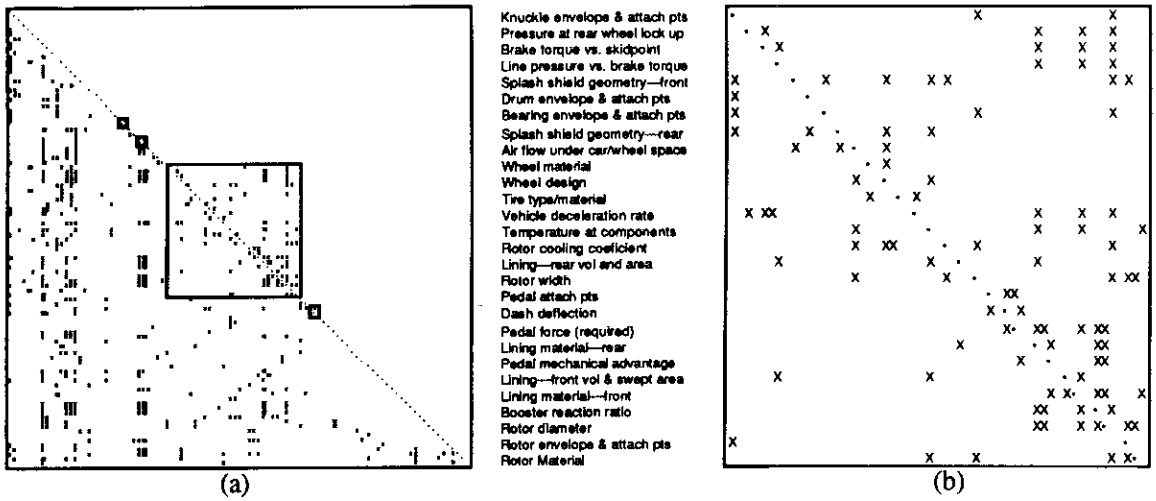



**Fig. 6.** Parameter-based design structure matrix for brake system design.

developed a simple four-level scheme which works well for quantifying the dependence of information inputs to each task:

| | |
|---|---|
| High | Information is required to begin the task. |
| Medium | Information is required to end the task. |
| Low | Information is needed only to check result compatibility. |
| Zero | No information is required. |

### Parametric-Level Design Description

In other design process modeling efforts, we use a very different approach, and have found very different results. To create a parameter-level description, we document the design process by interviewing engineers only (not their managers). We ask the designers which parameters must be known in order to set another design parameter. By documenting all of these precedence relationships, we develop a "complete" description of the design problem. When partitioned, the model identifies the flow of information required to develop the final design configuration from the customer requirements.

Figure 6 shows the design matrix model developed through detailed study of automotive brake systems. (This work was begun in a related study by Black [3] and was continued by Smith in our group.) For the brake system model, the design parameters have names like: rotor diameter, lining material, splash shield geometry, and booster travel. The full brake system model includes more than 100 design parameters, and this large matrix is shown in Fig. 6(a) to indicate the overall structure obtained after partitioning. The brake system matrix shows that about a third of the parameters can be determined rather sequentially beginning with the customer requirements. The difficult portion of the design problem is described by the large block of more than 30 tasks forming the center of the matrix. The remaining details of the design are worked out in the lower portion of the matrix, which involves little iteration.

The center block is magnified in Fig. 6(b) to display the coupling more clearly. These parameters must be set through an iterative process, which in actual practice utilizes several computer simulations to predict brake system performance. Unfortunately, this block also includes the prototyping and testing tasks, which take a considerable amount of time. (We find these tasks in the iterative design loop because certain design parameters require the test results in order to be finalized.) The engineers found the exposure of technical interactions in this case to be quite enlightening – many asked for a copy of the matrix diagram as it was taking shape, since this was the first time anyone had created a picture of the whole process. Similarly, managers appreciated the matrix model for its power to explain the complex flow of information within the iteration process. This model helped to launch our related work on modeling design iteration in which we analyze the mechanisms driving iterative problem solving [27].

### Hybrid Design Models

An important extension to this modeling work involves developing hybrid models – those which contain both task-level and parameter-level information. This is desirable because both the higher-level

task models and the lower-level parametric models are somewhat insufficient. The former ignore too many important technical details, while the latter lack the overall context. (For example, in the brake system model, we find that some of the most important redesign loops involve managerial decisions that are required when key design constraints cannot be met or when time and budget adjustments are needed.) We strive to create hybrid models which embody both types of features. Such a model enables the study of the important feedback marks lying far above the diagonal, linking the blocks of tightly coupled task activity.

We have found that tasks can often be described as sets of parameters to be determined. These parameters are usually coupled within each task. By using a hybrid model, the coupling between tasks becomes exposed in sufficient detail to consider redefining tasks by regrouping parameters into new tasks. This is simply accomplished by modeling the design process as existing tasks first, parameters second, and finally repartitioning the matrix into new task groups.

Creating new models and redefining the inter-block constraints may provide new opportunities for innovative design management. We call this designing the design process, or "meta-design", discussed in the following section.

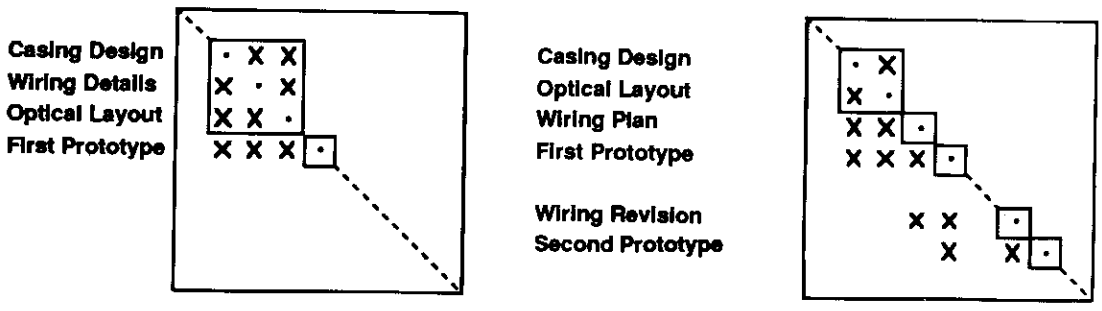
## Strategies for Designing Better Design Processes

Analyzing the technical structure of a development project can identify opportunities to improve the design process. This is particularly true if we are willing to modify the process by redefining tasks to alter their inherent coupling. To illustrate design improvement strategies, we present two conflicting approaches to consider: removing coupling versus adding coupling.

### Decoupling Tasks To Speed Design

A loosely coupled group of tasks can sometimes be split up into two or more smaller, more tightly coupled groups by *artificial decoupling*, which involves actually removing one or more task dependences (one or more marks) from the matrix. This can be accomplished in several ways, including the creation of an additional task to be performed earlier in the design procedure. The definition of this new task would require the parties associated with the removed dependence to agree ahead of time on the relevant task interfaces. Another approach to this artificial decoupling strategy is illustrated by the following example obtained by comparing the actual design procedures in two firms developing nearly identical products (an electro-mechanical instrument cluster) for the same customer (General Motors) [14].

Designers in one firm recognize three aspects of the product (the casing, wiring, and optics) to be so tightly coupled that they must be designed simultaneously, requiring lengthy negotiation (five to ten design iterations, taking up to six months) before enough detail can be settled to tool the first working prototype. The design structure matrix describing this procedure is shown in Fig. 7(a). The designers in the competing firm believe that a first prototype must be delivered much more quickly and that it is acceptable for the wiring inside such a prototype to be untidy (rather than hard tooled). They have developed the design procedure illustrated by Fig. 7(b), where the wiring is absent from the design iteration loop. Since only the casing and optical engineers are involved at first, the design is completed more quickly (in two iterations, taking only a few weeks) and the prototype is built with crude wiring. The final wiring layout is eventually completed for the second prototype. The wiring was artificially decoupled from the design in order to speed development.




(a) (b)

Fig. 7. Instrument design task matrices.

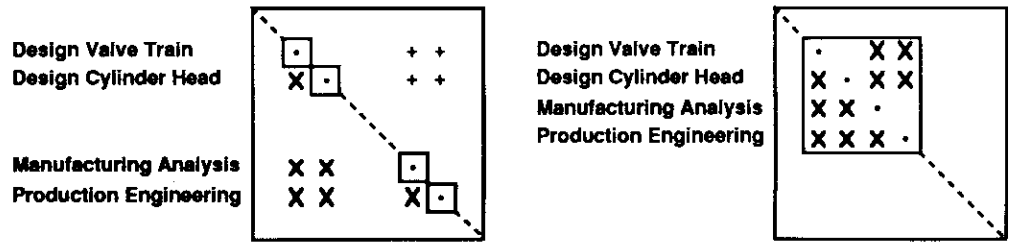



**Fig. 8.** Sequential and concurrent design procedures.

The artificial decoupling strategy of Fig. 7(b) clearly departs from the recommendations of concurrent engineering which are embodied in the procedure of Fig. 7(a). Decoupling is successful in this case because the coupling of the wiring detail is mostly unilateral. The feedback from wiring to the other two design tasks is less important. This is recognized and exploited by the faster design group, where the wiring engineers are given little opportunity to influence the earlier design stages. One potential criticism for this less coupled but faster design procedure is potentially inferior quality due to the absence of one important voice in the early phase of the design process. In fact, it is true that the quick but relatively poor wiring represents a lesser-quality prototype; however, we observed that this faster design firm ultimately achieves superior quality which we believe is due to two factors: First, the casing and optics designers are provided with some basic understanding of the wiring constraints. Second, the wiring is easily revised as necessary before the second prototype is built, whereas the slower design firm has greater difficulty revising their (hard-tooled) version.

### Increasing Coupling to Improve Design Quality

An increased coupling strategy is the essential basis of simultaneous engineering and design for manufacture (DFM). In the traditional (sequential) design process, depicted by the matrix in Fig. 8(a), the product designers would perform their design tasks somewhat independently from the manufacturing engineers. In the modern (concurrent) design process, Fig. 8(b), the practice of DFM mandates that these two activities be performed simultaneously. This is beneficial because the production expertise is brought into the early design stages (often causing much iteration), resulting in designs which are simpler to manufacture – higher quality design. However, the added coupling in the design process in fact slows product development considerably. Advocates of this philosophy would argue that overall design time can still be reduced because the need for later (more lengthy) iteration is therefore lessened. This is particularly true if the feedback from manufacturing engineering to design was indeed present in the original design procedure. This feedback is shown in Fig. 8(a) by the + marks, which depict redesign activity addressing the production problems which inevitably arise. Concurrent engineering has therefore both strengthened the relationship and rearranged the tasks, which appears to improve quality and may accelerate the project as well.

## A Developmental Process Strategy

It has become accepted that adding coupling in a design process is helpful to improve product quality because it can provide feedback of multiple perspectives to early design decisions. However, since this strategy usually does increase iteration, potentially causing the design process to take more time to execute, one must not implement this scheme to the fullest extreme; too much feedback could actually stall design progress. If the matrix were full of marks because every decision were allowed to directly influence every other decision, then the design procedure might involve unacceptably many iterations. A compromise must be made to optimize the tradeoff between reducing design time and improving design quality. To achieve this, tasks must be defined and arranged so that rapid iteration and task integration can be achieved.

Figure 9 depicts an effective design management strategy which represents a hybrid of the sequential and concurrent schemes discussed above. We documented this example while working with a major semiconductor company (Intel Corporation) to help them understand the efficiency of their development process [16]. The matrix shown in Fig. 9 includes several major iterative blocks representing the tightly coupled subsystems which are developed in truly concurrent fashion. Note that many of these concurrent activity blocks are overlapped, requiring a

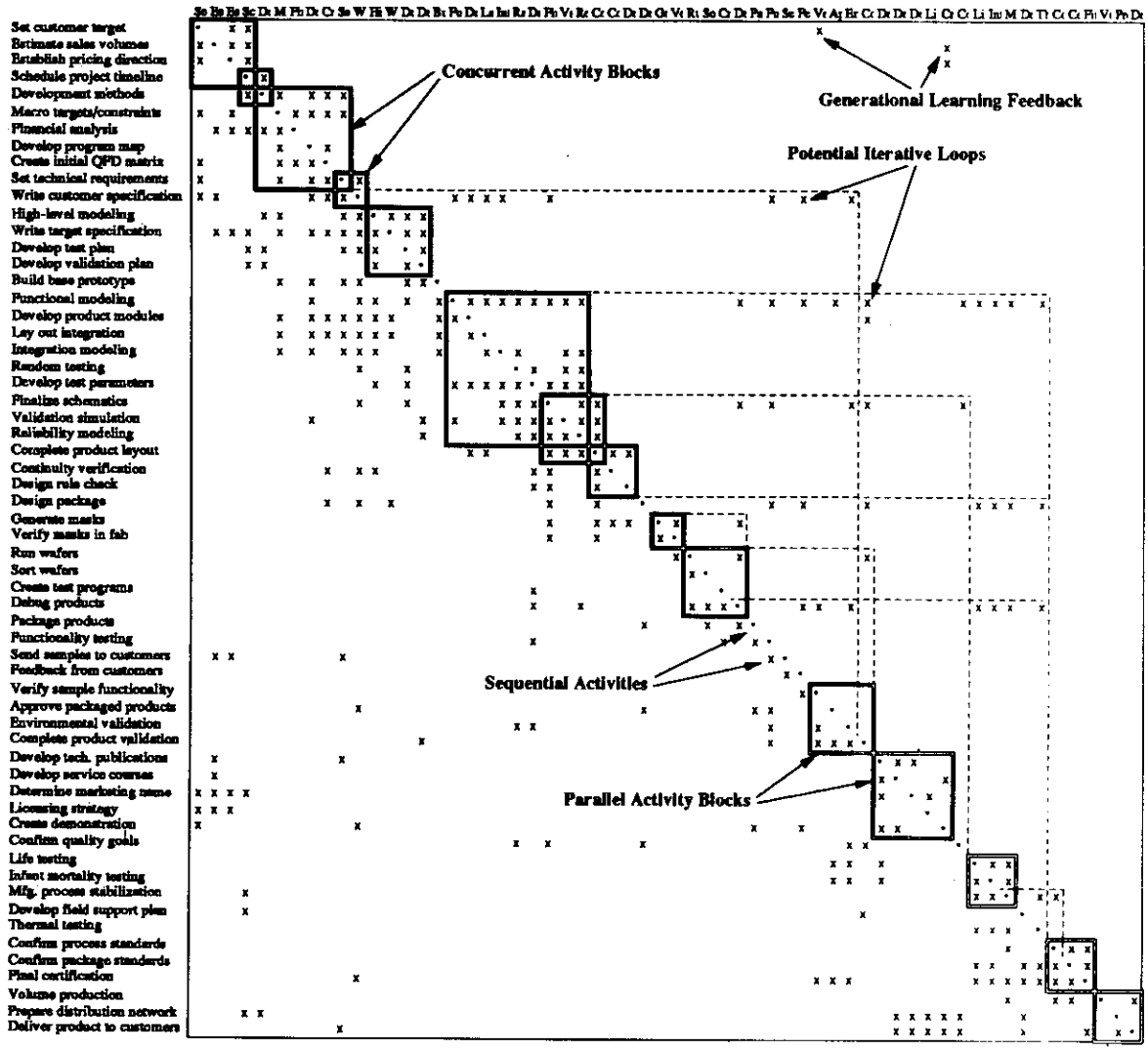



**Fig. 9.** Structured development process strategy (semiconductor design).

a great deal of coordination. The matrix also identifies which sets of activities can be executed in parallel and which should be attempted sequentially. Above the iterative subsystem development blocks lie the most important elements of this development procedure: the system-level feedback. The few feedback marks which are above the block diagonal partitioning drive the longer iterations among the sub-systems. Since these iterations may involve major development efforts (costly design *re*work), they must be managed very carefully – by using CAD tools to accelerate such rework, and by avoiding the need for this rework whenever possible. The third level of iteration shown in the matrix is what we call generational learning feedback. These marks at the far upper right represent the lessons learned later in the project that must be passed on to the next generation of the product since it is too late to make such changes to the current product.

We believe that this example demonstrates a carefully structured development process. In our experience with industrial projects, we have found such a matrix to be obtained through careful planning and reflection by the development team and its leaders. Dramatic improvements in development time require a sophisticated understanding of interrelationships and iteration drivers. To create a superior development process strategy, the design team leaders must take systematic steps toward improvement. Design managers can implement this scheme by following the plan outlined below.

1. Engage designers and engineers in a development process modeling activity. We have found that

model development requires approximately one month's effort for an engineer or manager familiar with the entire process (to create a matrix model with about fifty interacting tasks or parameters). This activity can sometimes be accelerated using surveys or team meetings instead of individual interviews. Using the matrix format to display the model builds group consensus and forms the basis for process analysis and coordination.

2. Find an appropriate set of major tasks into which the overall project may be divided. The best partitioning may not coincide with natural or traditional subsystems. Rather they may be found in other areas where many tasks are inherently tightly coupled owing to the underlying problem structure.

3. Facilitate design iteration within these very tightly coupled task blocks. This may require design automation tools, improved channels of communication, and/or changes in group membership or organization. When implemented properly, more iterations would actually be conducted in less time, focusing on the most important issues.

4. Allow many tasks to be performed in parallel when possible. This may be accomplished by encouraging all participants to identify where their needed information is generated and to collect those inputs as soon as they are available.

5. Remove some of the less important task couplings which might otherwise cause wasteful iteration. These may be very difficult to recognize and would generally be a matter of conflicting opinions, however, the leverage gained by streamlining a few tasks may be substantial if this allows many others to be more productive as well.

6. Most importantly, design managers must decide strategically where to place the important iteration drivers. Some of these longer feedback loops are essential to the current design. Others are for generational learning to improve future design efforts. The key feedback elements must be preserved and these loops should be shortened where possible by performing coupled tasks closely together.

7. Direct an ongoing effort to continuously modify and improve the matrix design process model. Solicit suggestions for improvement while spreading process understanding throughout the organization.

## Conclusion

We recognize that product development is difficult for several reasons: products can be technologically complex; a complete design procedure may involve millions of tasks; and all the tasks are coupled in some manner, making iteration an inherent characteristic of the design activity. We claim that the design process can be performed more successfully if it can be organized more sensibly.

An industrial product development process involves many interrelated engineering design procedures. The first step in improving such a process is to model and understand it. However, the design process has so far lacked convincing and effective models that permit analysis and systematic development of improvements. The design structure matrix and its associated process modeling effort is a step in this direction. It assumes that the basic elements of a design process are tasks that require input information, take time to execute, and produce decisions or output information for transfer to other tasks. It further assumes that a major route to process improvement, other than making each task more efficient, is to resequence individual tasks or groups of tasks so that required information is available sooner and available information is used sooner. An important extension to the resequencing paradigm is to redefine tasks by breaking them up into parameters and recombining these into new tasks. We observe that this can often be accomplished by careful scrutiny of the development procedure, which is facilitated by the design structure matrix.

Practical results of this approach take several forms. Even without applying algorithms for resequencing tasks, one can use the design structure matrix as a display of the existing design process or of the designers' view of the process. The matrix graphically displays all the existing information flows and makes it easy to see the difficulties in the form of coupling and unnecessary delays. Engineers can thus find their place in a large and dispersed activity. Pracht showed that even a simple directed graph was a powerful visual aid in decision making [18], and we have found that the matrix format reveals even more surprising features, including some problems that can be easily remedied. While this approach to design process improvement requires a detailed process model, this requirement is common to all past successful process improvement efforts. The resulting matrix can also be used as a management tool to redirect engineering effort to tasks involved in key iterations. Design reviews and progress assessments can be based on the matrix, with managers assuring that required information is transmitted, received and utilized in a

timely manner so that critical tasks can be accomplished as efficiently as possible.

In the future, our research and interaction with industry will produce computer tools that permit managers to find optimum ways of restructuring more complex design tasks, exposing problems and creating unique solutions that could not be found just by manually inspecting the matrix. One can also imagine the matrix augmented with designers' names, phone numbers, electronic mail addresses, datafile names and other information. Such information will make designers' work and communication more efficient and make the structure of engineering design databases more consistent with the needs of the design process and its information flows.

We have begun related work on modeling the solution of iterative design procedures such as the portions represented by the coupled blocks in design structure matrices [26, 27]. We are also investigating methods for overlapping nominally sequential tasks in product development by transferring preliminary information before the engineering work is completed [11]. We have been applying the methods outlined in this paper to document, study and improve design procedures in various firms with rather complicated product development practices (including semiconductor, telecommunication, aerospace and automotive industries).

## Acknowledgement

This research was conducted at the Massachusetts Institute of Technology. It was funded jointly by National Science Foundation, General Motors Corporation, and MIT Leaders for Manufacturing Program.

The four authors are affiliated with (respectively) MIT Sloan School of Management, MIT Center for Technology, Policy, and Industrial Development, University of Washington, and Motorola Inc.

## References

1. C. Alexander. *Notes on the Synthesis of Form*, Harvard University Press, Cambridge, MA, 1964.
2. T. Allen. *Managing the Flow of Technology: Technology Transfer and the Dissemination of Technological Information Within the R&D Organization*, MIT Press, Cambridge, MA, 1977.
3. T. A. Black. *A Systems Design Methodology Applied to Automotive Brake Design*, MIT, Masters Thesis, 1990.
4. D. M. Byrne and S. Taguchi. "The Taguchi Approach to Parameter Design", *Quality Progress*. December 1987, pp. 19–26.
5. K. B. Clark and T. Fujimoto. *Product Development Performance: Strategy, Organization, and Management in the World Auto Industry*, Harvard Business School Press, Boston, 1991.
6. J. W. Dean Jr and G. I. Susman. "Organizing for Manufacturable Design", *Harvard Business Review*. January– February 1989, pp. 28–36.
7. P. F. Drucker. "The Discipline of Innovation", *Harvard Business Review*. May–June 1985, pp. 67–72.
8. D. A. Gebala and S. D. Eppinger. *Methods for Analyzing Design Procedures*, ASME Conference on Design Theory and Methodology, Miami, September 1991, pp. 227–233.
9. D. M. Himmelblau. "Decomposition of Large Scale Systems, Part 1: Systems Composed of Lumped Parameter Elements", *Chemical Engineering Science*. vol. 21, 1966, pp. 425–438.
10. E. Kehat and M. Shacham. "Chemical Process Simulation Programs, Part 2: Partitioning and Tearing of System Flowsheets", *Process Technology International*. vol. 18, no. 3, March 1973, pp. 115–118.
11. V. Krishnan, S. D. Eppinger and D. E. Whitney. *A Model-Based Framework for Overlapping Product Development Activities*, MIT Sloan School of Management Working Paper, November 1993.
12. W. P. Ledet and D. M. Himmelblau. "Decomposition Procedures for the Solving of Large Scale Systems", *Advances in Chemical Engineering*. vol. 8, 1970, pp. 185–254.
13. D. A. Marca and C. L. McGowen. *SADT: Structural Analysis and Design Technique*, McGraw-Hill, New York, 1988.
14. D. A. Marshall. *Dynamic Benchmarking: A Comparative Study of Automotive Suppliers*, MIT, Masters Thesis, 1991.
15. J. L. Nevins and D. E. Whitney. *Concurrent Design of Products and Processes*, McGraw-Hill, New York, 1989.
16. S. M. Osborne. *Product Development Cycle Time Characterization Through Modeling of Process Iteration*, MIT, Masters Thesis, 1993.
17. S. L. Padula, C. Sandridge, R. T. Haftka and J. L. Walsh. "Demonstration of Decomposition and Optimization in the Design of Experimental Space Systems". In J.-F. M. Barthelemy, ed. *Recent Advances in Multidisciplinary Analysis*, NASA Langley Research Center, Hampton, VA, 1988, pp. 297–316.
18. W. E. Pracht. "Gismo: A Visual Problem-Structuring and Knowledge-Organization Tool", *IEEE Transactions on Systems, Man, and Cybernetics*. vol. SMC-16, no. 2, March–April 1986, pp. 265–270.
19. J. B. Quinn. "Managing Innovation: Controlled Chaos", *Harvard Business Review*. May–June 1985, pp. 73–84.
20. J. R. Rinderle and V. Krishnan. "Constraint Reasoning in Design", *International Conference on Design Theory and Methodology*. Chicago, September 1990.
21. J. R. Rinderle and N. P. Suh. "Measures of Functional Coupling in Design", *ASME Journal of Engineering for Industry*. November 1982, pp. 383–388.
22. J. L. Rogers. *DeMAID: A Design Manager's Aide for Intelligent Decomposition User's Guide*, NASA Technical Memorandum, 101575, March 1989.
23. J. L. Rogers and S. L. Padula. *An Intelligent Advisor for the Design Manager*, NASA Technical Memorandum, 101558, February 1989.
24. D. T. Ross. "Structured Analysis (SA): A Language for Communicating Ideas", *IEEE Transactions on Software Engineering*. vol. SE-3, no. 1, January 1977, pp. 16–34.
25. H. A. Simon. *The Sciences of the Artificial*, MIT Press, Cambridge, MA, 1970.

26. R. P. Smith and S. D. Eppinger. *A Predictive Model of Sequential Iteration in Engineering Design*, MIT Sloan School of Management Working Paper, no. 3160, rev. November 1991.
27. R. P. Smith and S. D. Eppinger. *Identifying Controlling Features of Engineering Design Iteration*, MIT Sloan School of Management Working Paper, no. 3348, rev. September 1992.
28. J. Sobieszczanski-Sobieski. *Multidisciplinary Optimization for Engineering Systems: Achievements and Potential*, NASA Technical Memorandum 101566, March 1989.
29. D. Sriram and M. L. Maher. "Representation and Use of Constraints in Structural Design", *AI in Engineering*. Springer-Verlag, Southampton, UK, April 1986.
30. D. V. Steward. "Partitioning and Tearing Systems of Equations", *SIAM Journal of Numerical Analysis*. ser. B, vol. 2, no. 2, 1965, pp. 345–365.
31. D. V. Steward. "The Design Structure System: A Method for Managing the Design of Complex Systems", *IEEE Transactions on Engineering Management*. vol. EM-28, no. 3, August 1981, pp. 71–74.
32. D. V. Steward. *Systems Analysis and Management: Structure, Strategy, and Design*, Petrocelli Books, New York, 1981.
33. N. P. Suh. *The Principles of Design*, Oxford University Press, New York, 1990.
34. G. J. Sussman and G. L. Steele. "Constraints – A Language for Expressing Almost-Hierarchical Descriptions", *Artificial Intelligence*. vol. 14, 1980, pp. 1–39.
35. E. von Hippel. *The Sources of Innovation*, Oxford University Press, New York, 1988.
36. J. N. Warfield. "Binary Matrices in System Modeling", *IEEE Transactions on Systems, Man, and Cybernetics*. vol. SMC-3, no. 5, September 1973, pp. 441–449.
37. D. E. Whitney. "Manufacturing By Design", *Harvard Business Review*. July–August 1988, pp. 83–91.
38. D. E. Whitney and M. Milley. "CADSYS: A New Approach to Computer-Aided Design", *IEEE Transactions on Systems, Man, and Cybernetics*. vol. SMC-4, no. 1, January 1974, pp. 50–58.
39. J. D. Wiest and F. K. Levy. *A Management Guide to PERT/CPM*, Prentice-Hall, Englewood Cliffs, New Jersey, 2nd edition, 1977.